

# Distributionally Robust Chance-Constrained Bin Packing

Yiling Zhang, Ruiwei Jiang, Siqian Shen  
Department of Industrial and Operations Engineering  
University of Michigan, Ann Arbor, MI, USA  
E-mail: {zyiling, ruiwei, siqian}@umich.edu

## Abstract

Chance-constrained bin packing problem allocates a set of items into bins and, for each bin, bounds the probability that the total weight of packed items exceeds the bin's capacity. Different from the stochastic programming approaches relying on full distributional information of the random item weights, we assume that only the information of the mean and covariance matrix is available. Accordingly, we consider distributionally robust chance-constrained bin packing (DCBP) models. Using two types of ambiguity sets, we equivalently reformulate the DCBP models as 0-1 second-order cone (SOC) programs. Furthermore, we exploit the submodularity of the 0-1 SOC constraints under special and general covariance matrices, and derive extended polymatroid inequalities to strengthen the 0-1 SOC formulations. We then incorporate these valid inequalities in a branch-and-cut algorithm for efficiently solving the DCBP models. Finally, we demonstrate the computational efficacy of our approaches and performance of DCBP solutions on test instances with diverse problem sizes, parameters, and item weight uncertainty.

*Key words:* chance-constrained binary program; distributionally robust optimization; conic integer program; submodularity; extended polymatroid

## 1 Introduction

The classical bin packing problem involves a list of  $J$  items and a list of  $I$  bins, where each item has a certain weight and each bin has a weight capacity. The deterministic bin packing problem aims to assign all  $J$  items to a minimum number of bins, while respecting the capacity of each bin. If we consider a slightly more general setting by introducing a cost for each assignment, then the deterministic bin packing problem is presented as

$$\min_{\mathbf{z}, \mathbf{y}} \quad \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} \quad (1a)$$

$$\text{s.t.} \quad y_{ij} \leq \rho_{ij} z_i \quad \forall i = 1, \dots, I, \quad j = 1, \dots, J \quad (1b)$$

$$\sum_{i=1}^I y_{ij} = 1 \quad \forall j = 1, \dots, J \quad (1c)$$

$$y_{ij}, z_i \in \{0, 1\} \quad \forall i = 1, \dots, I, \quad j = 1, \dots, J, \quad (1d)$$

$$\sum_{j=1}^J t_{ij} y_{ij} \leq T_i \quad \forall i = 1, \dots, I \quad (1e)$$

where  $c_i^z$  represents the cost of opening bin  $i$  and  $c_{ij}^y$  represents the cost of assigning item  $j$  to bin  $i$ . Meanwhile, we let binary variables  $z_i$  represent if bin  $i$  is opened (i.e.,  $z_i = 1$  if opened and  $z_i = 0$  otherwise), binary variables  $y_{ij}$  represent if item  $j$  is assigned to bin  $i$  (i.e.,  $y_{ij} = 1$  if assigned and  $y_{ij} = 0$  otherwise), parameters  $\rho_{ij}$  represent if we can assign item  $j$  to bin  $i$  (i.e.,  $\rho_{ij} = 1$  if we can and  $\rho_{ij} = 0$  otherwise), parameters  $t_{ij}$  represent the weight of item  $j$  accounts for in bin  $i$ , and parameters  $T_i$  represent the weight capacity of bin  $i$ . The objective function (1a) minimizes the total cost of opening bins and assigning items to bins. Constraints (1b) ensure that all items are assigned to open bins, constraints (1c) ensure that each item is assigned to one and only one bin, and constraints (1e) describe the capacity of each bin. The bin packing model (1) has a wide range of applications, including the cutting stock problem (Gilmore and Gomory, 1961), allocating surgeries to operating rooms (ORs) (Denton et al., 2010; Shylo et al., 2012), assigning computational jobs to servers in computing clouds (Shen and Wang, 2014; Wang and Shen, 2012), and scheduling crash tests on prototype vehicles (Reich et al., 2015).

In practice, the weights  $t_{ij}$  are often random when we decide the bin opening and item assignment. In this case, we denote the random weights  $\tilde{t}_{ij}$ , for all  $i = 1, \dots, I$ ,  $j = 1, \dots, J$ . For example, in surgery planning, the operational time limit of each OR (i.e.,  $T_i$ ) is usually deterministic, but the processing time of each surgery (i.e.,  $\tilde{t}_{ij}$ ) is usually random due to the variety of patients, surgical teams, and surgery characteristics. As a consequence, the capacity constraints (1e) are subject to random violations. An effective and convenient way of controlling constraint violations employs chance constraints. In a chance-constrained bin packing model, we attempt to satisfy each capacity constraint by at least a pre-specified probability, i.e.,

$$\mathbb{P}\left\{\sum_{j=1}^J \tilde{t}_{ij} y_{ij} \leq T_i\right\} \geq 1 - \alpha_i \quad \forall i = 1, \dots, I, \quad (2)$$

where  $\mathbb{P}$  represents the joint probability distribution of the random weights  $\{\tilde{t}_{ij} : j = 1, \dots, J\}$  and each  $\alpha_i$  represents an allowed risk tolerance of constraint violation that often takes a small value (e.g.,  $\alpha_i = 0.05$ ). For example, in surgery allocation, chance constraints (2) make sure that each OR will not go overtime with a large probability, offering an appropriate “end-of-the-day” guarantee.

Chance-constrained programming models are difficult to solve, mainly because the feasible region described by constraints (2) is non-convex in general (Prékopa, 2003). Nonetheless, promising special cases have been identified to recapture the convexity of chance-constrained models. In particular, if  $\{\tilde{t}_{ij} : j = 1, \dots, J\}$  are assumed to follow a Gaussian distribution with a known mean  $\mu_i$  and covariance matrix  $\Sigma_i$ , then the chance constraints (2) are equivalent to the second-order cone (SOC) constraints

$$\mu_i^\top y_i + \Phi^{-1}(1 - \alpha_i) \sqrt{y_i^\top \Sigma_i y_i} \leq T_i \quad \forall i = 1, \dots, I, \quad (3)$$

where  $\Phi(\cdot)$  represents the cumulative distribution function of the standard Gaussian distribution

and vector  $y_i := [y_{i1}, \dots, y_{iJ}]^\top$ . In this case, the chance-constrained bin packing model becomes a 0-1 SOC program and so it can be effectively solved by using the off-the-shelf solvers, such as GUROBI and CPLEX. In another promising stream of research, the probability distribution  $\mathbb{P}$  of  $\tilde{t}_{ij}$  is replaced by a finite-sample approximation, leading to a sample average approximation (SAA) of the chance-constrained model (Luedtke and Ahmed, 2008; Pagnoncelli et al., 2009). The SAA model is then recast as a mixed-integer linear program (MILP), on which many strong valid inequalities can be derived to accelerate the branch-and-cut algorithm (see, e.g., Luedtke et al., 2010; Küçükyavuz, 2012; Luedtke, 2014; Song et al., 2014; Liu et al., 2016).

A basic challenge of the chance-constrained approach is that the perfect knowledge of probability distribution  $\mathbb{P}$  may not be accessible. Under many circumstances, we only have a series of historical data that can be considered as samples taken from the true (while ambiguous) distribution. As a consequence, the solution obtained from a chance-constrained model can be sensitive to the choice of distribution  $\mathbb{P}$  we employ in (2) and hence perform poorly in out-of-sample tests. This phenomenon is often observed when solving stochastic programs and is called the *optimizer's curse* (Smith and Winkler, 2006). A natural way of addressing this curse is that, instead of a single estimate of  $\mathbb{P}$ , we employ a set of plausible probability distributions, termed the ambiguity set and denoted  $\mathcal{D}$ . Then, from a robust perspective, we ensure that chance constraints (2) hold valid with regard to all probability distributions belonging to  $\mathcal{D}$ , i.e.,

$$\inf_{\mathbb{P} \in \mathcal{D}} \mathbb{P} \left\{ \sum_{j=1}^J \tilde{t}_{ij} y_{ij} \leq T_i \right\} \geq 1 - \alpha_i \quad \forall i = 1, \dots, I, \quad (4)$$

and accordingly we call (4) *distributionally robust chance constraints* (DRCCs). In this paper, we investigate the distributionally robust chance-constrained bin packing (DCBP) model, a variant of the formulation (1), where capacity constraints (1e) are replaced with their distributionally robust counterpart (4). Without making the Gaussian assumption on  $\mathbb{P}$ , we show that DCBP is equivalent to a 0-1 SOC program when  $\mathcal{D}$  is characterized by the first two moments of  $\tilde{t}_{ij}$ . Furthermore, we exploit the (hidden) submodularity of the 0-1 SOC constraints. Accordingly, we derive extended polymatroid inequalities, whose separation can be efficiently done via a greedy algorithm. As demonstrated in extensive computational experiments, these valid inequalities significantly accelerate the branch-and-cut algorithm for solving DCBP. We also note that the branch-and-cut algorithm based on submodularity and extended polymatroid inequalities is general, and can be applied to accelerating the computation of general 0-1 SOC programs, but not only restricted to solving the ones for modeling DCBP in this paper.

The remainder of the paper is organized as follows. Section 2 reviews the literature in stochastic bin packing and the prior work related to the optimization techniques used in this paper. Section 3 presents two 0-1 SOC representations, respectively, for the DCBP models under two moment-based ambiguity sets from the literature. Section 4 utilizes submodularity, lifting, and bin-packing structures to derive valid inequalities to strengthen the 0-1 SOC formulations. Section 5 demonstrates the computational efficacy of our approaches and compare the out-of-sample performance of differ-

ent DCBP models based on test instances with diverse problem sizes, parameters, and item weight uncertainty. Section 6 summarizes the paper and discusses future research directions.

## 2 Prior Work

The stochastic bin packing models find a wide range of applications, ranging from surgery planning to job scheduling in data centers to bandwidth allocation in high-speed networks. Denton et al. (2010) optimized surgery block allocation to ORs, by minimizing the total cost of opening the ORs plus the expected penalty cost of having overtime in each OR. They solved a two-stage stochastic binary integer program via an integer L-shaped method, based on finite samples of the random surgery durations. Shylo et al. (2012) was among the first to use chance constraints for restricting the overtime in ORs. They assumed that the surgery durations follow a multivariate Gaussian distribution, and reformulated the chance-constrained model as an equivalent semidefinite program (SDP) based on convex analysis of probabilistic constraints dating back to Prékopa (1970). Deng et al. (2016) provided a summary of recent papers that utilized variants of stochastic bin packing for surgery planning optimization. Shen and Wang (2014) illustrated other stochastic bin packing applications for managing the quality of service and enhancing operational efficiency of server consolidation in cloud computing systems. Kleinberg et al. (2000) related stochastic bin packing to the problem of allocating bandwidth for bursty connections in high-speed networks, and proposed approximation algorithms for its chance-constrained variant.

Chance-constrained bin packing problems are computationally challenging, largely because (i) the non-convexity of chance constraints and (ii) the binary restrictions on decision variables. To address challenges (i)–(ii), the majority of existing literature requires full distributional knowledge of the random item weights, and applies the SAA approach to approximate the chance-constrained bin packing problem as an MILP. For example, Song et al. (2014) considered a generic chance-constrained binary packing problem given finite samples of the random item weights, and derived lifted cover inequalities for improving the computation. Zhang et al. (2015) applied branch-and-price and column generation approaches to optimize an SAA-based MILP for chance-constrained bin packing. For generic chance-constrained programs, the SAA model and the valid inequalities for its MILP formulation have been well studied in the literature (see, e.g., Luedtke and Ahmed, 2008; Pagnoncelli et al., 2009; Luedtke et al., 2010; Küçükyavuz, 2012; Luedtke, 2014; Liu et al., 2016). As compared to the existing work, this paper waives the assumption of full distributional information and only relies on the first two moments of the uncertainty.

Based on a bandwidth allocation problem, Kleinberg et al. (2000) pointed out the relations between stochastic bin packing and stochastic knapsack problems. In fact, the chance constraints in binary knapsack problems are equivalent to those in bin pack problems. Kosuch and Lisser (2010), Goyal and Ravi (2010), and Han et al. (2016) considered knapsack chance constraints with Gaussian distributed random item weights. In particular, Kosuch and Lisser (2010) recast the chance constraint as an SOC constraint. Goyal and Ravi (2010) provided a polynomial-time approximation algorithm for the chance-constrained binary knapsack problem. Recently, Han et al.

(2016) approximated the chance constraint from a robust optimization perspective and provided efficient solution approaches for the approximation. Different from these work, this paper derives an SOC representation of the chance constraint without assuming Gaussian random item weights. Meanwhile, we study solution approaches that achieve the global optimality of DCBP.

Distributionally robust optimization (DRO) has received growing attention in recent years. It introduces effective modeling and computational approaches for handling ambiguous distributions of random variables in stochastic programming problems by using available distributional information. The concept was first introduced by Scarf et al. (1958) as a minimax stochastic program for managing inventory under uncertain demand with only moment information known to a manager. Moment information (e.g., mean and covariance) has been widely used for building ambiguity sets in various DRO models (see, e.g., Bertsimas et al., 2010; Delage and Ye, 2010; Wiesemann et al., 2014). By using moment-based ambiguity sets, El Ghaoui et al. (2003), Calafiore and El Ghaoui (2006), Wagner (2008), Chen et al. (2010), Zymmler et al. (2013), Cheng et al. (2014), and Jiang and Guan (2016) derived exact reformulations and/or approximations for DRCCs, often in the form of SDPs. In special cases, e.g., when the first two moments are *exactly* matched in the ambiguity set, these SDPs can further be simplified as SOC programs that are even more computationally tractable. While many existing ambiguity sets exactly match the first two moments of uncertainty (see, e.g., El Ghaoui et al., 2003; Wagner, 2008; Zymmler et al., 2013), Delage and Ye (2010) proposed a data-driven approach to construct an ambiguity set that can model moment estimation errors. In this paper, we consider both types of ambiguity sets. To the best of our knowledge, for the first time, we provide an SOC representation of DRCCs using the general ambiguity set proposed by Delage and Ye (2010).

DRO has received much less attention in discrete optimization problems, possibly due to the difficulty of solving 0-1 nonlinear programs. For example, none of the off-the-shelf solvers can handle 0-1 SDPs, which often arise from discrete optimization problems with DRCCs. To the best of our knowledge, our results on chance constraints are most related to Cheng et al. (2014) that studied DRCCs in the binary knapsack problem and derived 0-1 SDP reformulations. As compared to their work, this paper investigates a different ambiguity set and derives a 0-1 SOC representation. Furthermore, we derive valid inequalities to solve the 0-1 SOC reformulation to global optimality, instead of considering an SDP relaxation of the binary restrictions as in Cheng et al. (2014).

The main contributions of the paper are three-fold. First, using the ambiguity set proposed by Delage and Ye (2010), we equivalently reformulate the DCBP model as a 0-1 SOC program that can readily be solved by the off-the-shelf solvers. Second, we derive a sufficient condition for the submodularity of 0-1 SOC constraints under a general covariance matrix. We also provide an efficient way of finding “optimal” submodular approximations of the 0-1 SOC constraints. Additionally, we show that general 0-1 SOC constraints become submodular in a lifted space. Based on these submodular properties, we derive extended polymatroid inequalities to accelerate solving DCBP. Third, we conduct extensive numerical studies to demonstrate the computational efficacy of our solution approaches and the out-of-sample performance of DCBP.

### 3 DCBP Models and Reformulations

We study the DCBP model under two alternatives of ambiguity set  $\mathcal{D}$  based on the first two moments of  $\tilde{t}_{ij}$ . The first ambiguity set, denoted  $\mathcal{D}_1$ , exactly matches the mean and covariance matrix of  $\tilde{t}_{ij}$ . In contrast, the second ambiguity set, denoted  $\mathcal{D}_2$ , considers the estimation errors of sample mean and sample covariance matrix (see Delage and Ye, 2010). In Section 3.1, we introduce these two ambiguity sets and their calibration based on historical data. In Section 3.2, we derive SOC representations of DRCC (4) under  $\mathcal{D}_1$  and  $\mathcal{D}_2$ , respectively. While the former case (i.e., (4) under  $\mathcal{D}_1$ ) has been well studied (see, e.g., El Ghaoui et al., 2003; Calafiore and El Ghaoui, 2006; Zymler et al., 2013), to the best of our knowledge, this is the first work to show the SOC representation of the latter case (i.e., (4) under  $\mathcal{D}_2$ ).

#### 3.1 Ambiguity Sets

For each bin  $i = 1, \dots, I$ , we let vector  $\tilde{t}_i := [\tilde{t}_{i1}, \dots, \tilde{t}_{iJ}]^\top$  represent the random weight of each item in bin  $i$ . Suppose that a series of independent historical data samples  $\{\tilde{t}_i^n\}_{n=1}^N$  are drawn from the true probability distribution  $\mathbb{P}$  of  $\tilde{t}_i$ . Then, the first two moments of  $\tilde{t}_i$  can be estimated by the sample mean and sample covariance matrix

$$\mu_i = \frac{1}{N} \sum_{n=1}^N \tilde{t}_i^n, \quad \Sigma_i = \frac{1}{N} \sum_{n=1}^N (\tilde{t}_i^n - \mu_i)(\tilde{t}_i^n - \mu_i)^\top.$$

Throughout this paper, we assume that both  $\Sigma_i$  and the true covariance matrix of  $\tilde{t}_i$  are positive definite. As promised by the law of large numbers, as the data size  $N$  grows,  $\mu_i$  and  $\Sigma_i$  converge to the true mean and true covariance matrix of  $\tilde{t}_i$ , respectively. Hence, when  $N$  takes a large value, a natural choice of the ambiguity set consists of all probability distributions that match the sample moments  $\mu_i$  and  $\Sigma_i$ , i.e.,

$$\mathcal{D}_1(\mu, \Sigma) = \left\{ \mathbb{P} \in \mathcal{M}_+^J : \begin{array}{l} \mathbb{E}_{\mathbb{P}}[\tilde{t}_i] = \mu_i, \\ \mathbb{E}_{\mathbb{P}}[(\tilde{t}_i - \mu_i)(\tilde{t}_i - \mu_i)^\top] = \Sigma_i, \quad \forall i = 1, \dots, I \end{array} \right\},$$

where  $\mathcal{M}_+^J$  represents the set of all probability distributions on  $\mathbb{R}^J$ ,  $\mu$  represents the collection of  $\mu_i$ , and  $\Sigma$  represents the collection of  $\Sigma_i$  for all  $i = 1, \dots, I$ . For notation brevity, we use  $\mathcal{D}_1(\mu, \Sigma)$  and  $\mathcal{D}_1$  interchangeably without necessarily specifying the means and covariance matrices.

Under many circumstances, however, the historical data can be inadequate. With a small  $N$ , there may exist considerable estimation errors in  $\mu_i$  and  $\Sigma_i$ , which brings a layer of “moment ambiguity” into  $\mathcal{D}_1$  and adds to the existing distributional ambiguity of  $\mathbb{P}$ . To address the moment ambiguity and take into account the estimation errors, Delage and Ye (2010) proposed an alternative ambiguity set

$$\mathcal{D}_2 = \left\{ \mathbb{P} \in \mathcal{M}_+^J : \begin{array}{l} (\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i)^\top \Sigma_i^{-1} (\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i) \leq \gamma_1, \\ \mathbb{E}_{\mathbb{P}}[(\tilde{t}_i - \mu_i)(\tilde{t}_i - \mu_i)^\top] \preceq \gamma_2 \Sigma_i, \quad \forall i = 1, \dots, I \end{array} \right\},$$

where  $\gamma_1 > 0$  and  $\gamma_2 > \max\{\gamma_1, 1\}$  represent two given parameters. Set  $\mathcal{D}_2$  designates that (i) the true mean of  $\tilde{t}_i$  is within an ellipsoid centered at  $\mu_i$ , and (ii) the true covariance matrix of  $\tilde{t}_i$  is bounded from above by  $\gamma_2 \Sigma_i - (\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i)(\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i)^\top$  (note that  $\mathbb{E}_{\mathbb{P}}[(\tilde{t}_i - \mu_i)(\tilde{t}_i - \mu_i)^\top] = \mathbb{E}_{\mathbb{P}}[(\tilde{t}_i - \mathbb{E}_{\mathbb{P}}[\tilde{t}_i])(\tilde{t}_i - \mathbb{E}_{\mathbb{P}}[\tilde{t}_i])^\top] + (\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i)(\mathbb{E}_{\mathbb{P}}[\tilde{t}_i] - \mu_i)^\top$ ). Meanwhile, Delage and Ye (2010) offered a rigorous guideline for selecting  $\gamma_1$  and  $\gamma_2$  values (see Theorem 2 in Delage and Ye (2010)) so that  $\mathcal{D}_2$  includes the true distribution of  $\tilde{t}_i$  with a high confidence level. In practice, we can select the values of  $\gamma_1$  and  $\gamma_2$  via cross validation. For example, we can divide the  $N$  data points into two halves. We estimate sample moments  $(\mu_i^1, \Sigma_i^1)$  based on the first half of the data and  $(\mu_i^2, \Sigma_i^2)$  based on the second half. Then, we characterize  $\mathcal{D}_2$  based on  $(\mu_i^1, \Sigma_i^1)$ , and select  $\gamma_1$  and  $\gamma_2$  such that probability distributions with moments  $(\mu_i^2, \Sigma_i^2)$  belong to  $\mathcal{D}_2$ .

### 3.2 SOC Representations of the DRCC

Now we derive SOC representations of DRCC (4) for all  $i = 1, \dots, I$ . For notation brevity, we define vector  $y := [y_{i1}, \dots, y_{iJ}]$  and omit the subscript  $i$  throughout this section. First, we review the celebrated SOC representation of DRCC (4) under ambiguity set  $\mathcal{D}_1$  in the following theorem.

**Theorem 1.** (Adapted from El Ghaoui et al. (2003), also see Wagner (2008)) The DRCC (4) with  $\mathcal{D} = \mathcal{D}_1$  is equivalent to the following SOC constraint:

$$\mu^\top y + \sqrt{\frac{1-\alpha}{\alpha}} \sqrt{y^\top \Sigma y} \leq T. \quad (5)$$

Theorem 1 shows that we can recapture the convexity of chance constraints (2) by employing ambiguity set  $\mathcal{D}_1$  to model the  $\tilde{t}$  uncertainty. Perhaps more surprisingly, in this case, the convex feasible region characterized by DRCC (4) is SOC representable. It follows that the continuous relaxation of the DCBP model is an SOC program, one of the most computationally tractable nonlinear programs nowadays.

Next, we show that DRCC (4) under the ambiguity set  $\mathcal{D}_2$  is also SOC representable. This implies that the computational complexity of the DCBP remains the same even if we take the moment ambiguity into account. We present the main result of this section in the following theorem.

**Theorem 2.** DRCC (4) with  $\mathcal{D} = \mathcal{D}_2$  is equivalent to

$$\mu^\top y + \left( \sqrt{\gamma_1} + \sqrt{\left( \frac{1-\alpha}{\alpha} \right) (\gamma_2 - \gamma_1)} \right) \sqrt{y^\top \Sigma y} \leq T \quad (6a)$$

if  $\gamma_1/\gamma_2 \leq \alpha$ , and is equivalent to

$$\mu^\top y + \sqrt{\frac{\gamma_2}{\alpha}} \sqrt{y^\top \Sigma y} \leq T \quad (6b)$$

if  $\gamma_1/\gamma_2 > \alpha$ .

We prove Theorem 2 in two steps. In the first step, we project the random vector  $\tilde{t}$  and its ambiguity set  $\mathcal{D}_2$  from  $\mathbb{R}^J$  to the real line, i.e.,  $\mathbb{R}$ . This simplifies DRCC (4) as involving a

one-dimensional random variable. In the second step, we derive optimal (i.e., worst-case) mean and covariance matrix in  $\mathcal{D}_2$  that attain the worst-case probability bound in (4). We then reuse Theorem 1 to finish the representation. We present the first step of the proof in the following lemma.

**Lemma 1.** Let  $\tilde{s}$  be a random vector in  $\mathbb{R}^J$  and  $\tilde{\xi}$  be a random variable in  $\mathbb{R}$ . For a given  $y \in \mathbb{R}^J$ , define ambiguity sets  $\mathcal{D}_{\tilde{s}}$  and  $\mathcal{D}_{\tilde{\xi}}$  as

$$\mathcal{D}_{\tilde{s}} = \left\{ \mathbb{P} \in \mathcal{M}_+^J : \mathbb{E}_{\mathbb{P}}[\tilde{s}]^\top \Sigma^{-1} \mathbb{E}_{\mathbb{P}}[\tilde{s}] \leq \gamma_1, \quad \mathbb{E}_{\mathbb{P}}[\tilde{s}\tilde{s}^\top] \preceq \gamma_2 \Sigma \right\} \quad (7a)$$

and

$$\mathcal{D}_{\tilde{\xi}} = \left\{ \mathbb{P} \in \mathcal{M}_+^1 : |\mathbb{E}_{\mathbb{P}}[\tilde{\xi}]| \leq \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}, \quad \mathbb{E}_{\mathbb{P}}[\tilde{\xi}^2] \leq \gamma_2 (y^\top \Sigma y) \right\}. \quad (7b)$$

Then, for any Borel measurable function  $f : \mathbb{R}^J \rightarrow \mathbb{R}$ , we have

$$\inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{f(y^\top \tilde{s}) \leq 0\} = \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{f(\tilde{\xi}) \leq 0\}.$$

*Proof.* We first show that  $\inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{f(y^\top \tilde{s}) \leq 0\} \geq \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{f(\tilde{\xi}) \leq 0\}$ . Pick a  $\mathbb{P} \in \mathcal{D}_{\tilde{s}}$ , and let  $\tilde{s}$  denote the corresponding random vector and  $\tilde{\xi} = y^\top \tilde{s}$ . It follows that

$$\begin{aligned} \mathbb{E}_{\mathbb{P}}[\tilde{\xi}] &= y^\top \mathbb{E}_{\mathbb{P}}[\tilde{s}] \\ &\leq \max_{s: s^\top \Sigma^{-1} s \leq \gamma_1} y^\top s \\ &= \max_{z: \|z\|_2 \leq \sqrt{\gamma_1}} (\Sigma^{1/2} y)^\top z = \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}, \end{aligned} \quad (8a)$$

where inequality (8a) is because  $\mathbb{E}_{\mathbb{P}}[\tilde{s}]^\top \Sigma^{-1} \mathbb{E}_{\mathbb{P}}[\tilde{s}] \leq \gamma_1$ . Similarly, we have  $\mathbb{E}_{\mathbb{P}}[\tilde{\xi}] \geq -\sqrt{\gamma_1} \sqrt{y^\top \Sigma y}$ . Meanwhile, note that

$$\begin{aligned} \mathbb{E}_{\mathbb{P}}[\tilde{\xi}^2] &= y^\top \mathbb{E}_{\mathbb{P}}[\tilde{s}\tilde{s}^\top] y \\ &\leq y^\top (\gamma_2 \Sigma) y = \gamma_2 (y^\top \Sigma y), \end{aligned} \quad (8b)$$

where inequality (8b) is because  $\mathbb{E}_{\mathbb{P}}[\tilde{s}\tilde{s}^\top] \preceq \gamma_2 \Sigma$ . Hence, the probability distribution of  $\tilde{\xi}$  belongs to  $\mathcal{D}_{\tilde{\xi}}$ . It follows that  $\inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{f(y^\top \tilde{s}) \leq 0\} \geq \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{f(\tilde{\xi}) \leq 0\}$ .

Second, we show that  $\inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{f(y^\top \tilde{s}) \leq 0\} \leq \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{f(\tilde{\xi}) \leq 0\}$ . Pick a  $\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}$ , and let  $\tilde{\xi}$  denote the corresponding random variable and  $\tilde{s} = [\tilde{\xi}/(y^\top \Sigma y)] \Sigma y$ . It follows that

$$\begin{aligned} \mathbb{E}_{\mathbb{P}}[\tilde{s}]^\top \Sigma^{-1} \mathbb{E}_{\mathbb{P}}[\tilde{s}] &= \mathbb{E}_{\mathbb{P}}[\tilde{\xi}]^2 \frac{y^\top \Sigma}{y^\top \Sigma y} \Sigma^{-1} \frac{\Sigma y}{y^\top \Sigma y} = \frac{\mathbb{E}_{\mathbb{P}}[\tilde{\xi}]^2}{y^\top \Sigma y} \\ &\leq \frac{\gamma_1 y^\top \Sigma y}{y^\top \Sigma y} = \gamma_1, \end{aligned} \quad (8c)$$



where inequality (8c) is because  $|\mathbb{E}_{\mathbb{P}}[\tilde{\xi}]| \leq \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}$ . Meanwhile, note that

$$\begin{aligned} \mathbb{E}_{\mathbb{P}}[\tilde{s}\tilde{s}^\top] &= \mathbb{E}_{\mathbb{P}}\left[\tilde{\xi}^2 \frac{\Sigma y}{y^\top \Sigma y} \frac{y^\top \Sigma}{y^\top \Sigma y}\right] \\ &= \mathbb{E}_{\mathbb{P}}[\tilde{\xi}^2] \frac{(\Sigma y)(\Sigma y)^\top}{(y^\top \Sigma y)^2} \\ &\preceq \gamma_2(y^\top \Sigma y) \frac{(\Sigma y)(\Sigma y)^\top}{(y^\top \Sigma y)^2} \end{aligned} \quad (8d)$$

$$\preceq \gamma_2(y^\top \Sigma y) \frac{(y^\top \Sigma y)\Sigma}{(y^\top \Sigma y)^2} = \gamma_2 \Sigma, \quad (8e)$$

where inequality (8d) is because  $\mathbb{E}_{\mathbb{P}}[\tilde{\xi}^2] \leq \gamma_2(y^\top \Sigma y)$  and inequality (8e) is because  $(\Sigma y)(\Sigma y)^\top \preceq (y^\top \Sigma y)\Sigma$ , which holds because for all  $z \in \mathbb{R}^J$ ,

$$\begin{aligned} z^\top (\Sigma y)(\Sigma y)^\top z &= \left[ (\Sigma^{1/2} z)^\top (\Sigma^{1/2} y) \right]^2 \\ &\leq \|\Sigma^{1/2} z\|^2 \|\Sigma^{1/2} y\|^2 \quad (\text{Cauchy-Schwarz inequality}) \\ &= (y^\top \Sigma y)(z^\top \Sigma z) \\ &= z^\top [(y^\top \Sigma y)\Sigma] z. \end{aligned}$$

Hence, the probability distribution of  $\tilde{s}$  belongs to  $\mathcal{D}_{\tilde{s}}$ . It follows that  $\inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{f(y^\top \tilde{s}) \leq 0\} \leq \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{f(\tilde{\xi}) \leq 0\}$  because  $\tilde{\xi} = y^\top \tilde{s}$ , and the proof is completed.  $\square$

We are now ready to finish the proof of Theorem 2.

*Proof of Theorem 2:* First, we define random vector  $\tilde{s} = \tilde{t} - \mu$ , random variable  $\tilde{\xi} = y^\top \tilde{s}$ , constant  $b = T - \mu^\top y$ , and set  $S$  such that

$$S = \{(\mu_1, \sigma_1) \in \mathbb{R} \times \mathbb{R}_+ : |\mu_1| \leq \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}, \mu_1^2 + \sigma_1^2 \leq \gamma_2 y^\top \Sigma y\}.$$

It follows that

$$\begin{aligned} \inf_{\mathbb{P} \in \mathcal{D}_2} \mathbb{P}\{\tilde{t}^\top y \leq T\} &= \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{s}}} \mathbb{P}\{y^\top \tilde{s} \leq b\} \\ &= \inf_{\mathbb{P} \in \mathcal{D}_{\tilde{\xi}}} \mathbb{P}\{\tilde{\xi} \leq b\} \end{aligned} \quad (9a)$$

$$= \inf_{(\mu_1, \sigma_1) \in S} \inf_{\mathbb{P} \in \mathcal{D}_1(\mu_1, \sigma_1^2)} \mathbb{P}\{\tilde{\xi} \leq b\}, \quad (9b)$$

where  $\mathcal{D}_{\tilde{s}}$  and  $\mathcal{D}_{\tilde{\xi}}$  are defined in (7a) and (7b), respectively, equality (9a) follows from Lemma 1, and equality (9b) decomposes the optimization problem in (9a) into two layers: the outer layer searches for the optimal (i.e., worst-case) mean and covariance, while the inner layer computes the worst-case probability bound under the given mean and covariance. For the inner layer, based on

Theorem 1, we have

$$\inf_{\mathbb{P} \in \mathcal{D}_1(\mu_1, \sigma_1^2)} \mathbb{P}\{\tilde{\xi} \leq b\} = \begin{cases} \frac{(b-\mu_1)^2}{\sigma_1^2 + (b-\mu_1)^2}, & \text{if } b \geq \mu_1, \\ 0, & \text{o.w.} \end{cases}$$

As DRCC (4) states that  $\inf_{\mathbb{P} \in \mathcal{D}_2} \mathbb{P}\{\tilde{t}^\top y \leq T\} \geq 1 - \alpha > 0$ , we can assume  $b \geq \mu_1$  for all  $(\mu_1, \sigma_1) \in S$  without loss of generality. That is,

$$b \geq \max_{(\mu_1, \sigma_1) \in S} \mu_1 = \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}.$$

It follows that

$$\begin{aligned} \inf_{\mathbb{P} \in \mathcal{D}_2} \mathbb{P}\{\tilde{t}^\top y \leq T\} &= \inf_{(\mu_1, \sigma_1) \in S} \frac{(b - \mu_1)^2}{\sigma_1^2 + (b - \mu_1)^2} \\ &= \inf_{(\mu_1, \sigma_1) \in S} \frac{1}{\left(\frac{\sigma_1}{b - \mu_1}\right)^2 + 1}. \end{aligned} \quad (9c)$$

Note that the objective function value in (9c) decreases as  $\sigma_1/(b - \mu_1)$  increases. Hence, (9c) shares optimal solutions with the following optimization problem:

$$\inf_{(\mu_1, \sigma_1) \in S} -\left(\frac{\sigma_1}{b - \mu_1}\right). \quad (9d)$$

The feasible region of problem (9d) is depicted in the shaded area of Figure 1. Furthermore, we

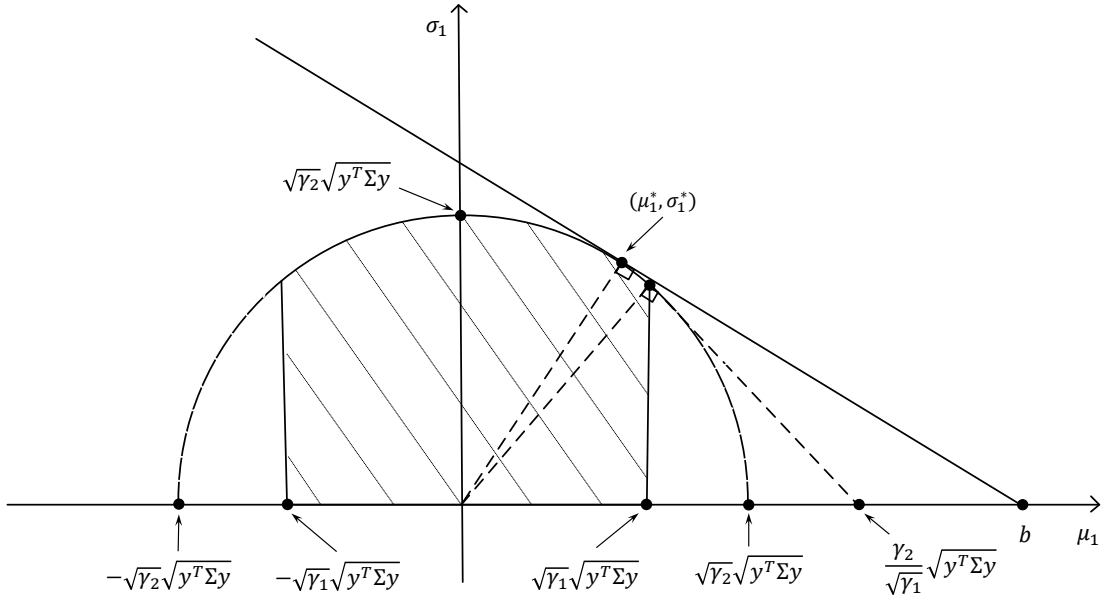


Figure 1: Graphical Solution of Problem (9d)

note that the objective function of (9d) equals to the slope of the straight line connecting points

$(b, 0)$  and  $(\mu_1, \sigma_1)$  (see Figure 1 for an example). It follows that an optimal solution  $(\mu_1^*, \sigma_1^*)$  to problem (9d), and so to problem (9c), lies in one of the following two cases:

**Case 1.** If  $\sqrt{\gamma_1} \sqrt{y^\top \Sigma y} \leq b \leq (\gamma_2 / \sqrt{\gamma_1}) \sqrt{y^\top \Sigma y}$ , then  $\mu_1^* = \sqrt{\gamma_1} \sqrt{y^\top \Sigma y}$  and  $\sigma_1^* = \sqrt{\gamma_2 - \gamma_1} \sqrt{y^\top \Sigma y}$ .

**Case 2.** If  $b > (\gamma_2 / \sqrt{\gamma_1}) \sqrt{y^\top \Sigma y}$ , then  $\mu_1^* = (\gamma_2 y^\top \Sigma y) / b$  and  $\sigma_1^* = \sqrt{\gamma_2 y^\top \Sigma y - (\gamma_2 y^\top \Sigma y)^2 / b^2}$ .

By denoting  $\kappa(b, y) = \frac{b}{\sqrt{y^\top \Sigma y}}$ , we have

$$\inf_{\mathbb{P} \in \mathcal{D}_2} \mathbb{P}\{\tilde{t}^\top y \leq T\} = \begin{cases} \frac{1}{\left(\frac{\sqrt{\gamma_2 - \gamma_1}}{\kappa(b, y) - \sqrt{\gamma_1}}\right)^2 + 1}, & \text{if } \sqrt{\gamma_1} \leq \kappa(b, y) \leq \frac{\gamma_2}{\sqrt{\gamma_1}}, \\ \frac{\left(\kappa(b, y) - \frac{1}{\kappa(b, y)}\right)^2}{\kappa(b, y)^2 - 1}, & \text{if } \kappa(b, y) > \frac{\gamma_2}{\sqrt{\gamma_1}}. \end{cases} \quad (9e)$$

Second, based on (9e), the DRCC  $\inf_{\mathbb{P} \in \mathcal{D}_2} \mathbb{P}\{\tilde{t}^\top y \leq T\} \geq 1 - \alpha$  has the following representations:

$$\text{DRCC} \Leftrightarrow \begin{cases} \frac{b}{\sqrt{y^\top \Sigma y}} \geq \sqrt{\gamma_1} + \sqrt{\left(\frac{1-\alpha}{\alpha}\right)(\gamma_2 - \gamma_1)}, & \text{if } \sqrt{\gamma_1} \leq \frac{b}{\sqrt{y^\top \Sigma y}} \leq \frac{\gamma_2}{\sqrt{\gamma_1}}, \\ \frac{b}{\sqrt{y^\top \Sigma y}} \geq \sqrt{\frac{\gamma_2}{\alpha}}, & \text{if } \frac{b}{\sqrt{y^\top \Sigma y}} > \frac{\gamma_2}{\sqrt{\gamma_1}}. \end{cases}$$

It follows that DRCC (4) is equivalent to an SOC constraint by discussing the following two cases:

**Case 1.** If  $\gamma_1 / \gamma_2 \leq \alpha$ , then  $\gamma_2 / \sqrt{\gamma_1} \geq \sqrt{\gamma_1} + \sqrt{[(1-\alpha)/\alpha](\gamma_2 - \gamma_1)}$  and  $\gamma_2 / \sqrt{\gamma_1} \geq \sqrt{\gamma_2 / \alpha}$ . It follows that DRCC (4) is equivalent to  $b / \sqrt{y^\top \Sigma y} \geq \sqrt{\gamma_1} + \sqrt{[(1-\alpha)/\alpha](\gamma_2 - \gamma_1)}$ .

**Case 2.** If  $\gamma_1 / \gamma_2 > \alpha$ , then  $\gamma_2 / \sqrt{\gamma_1} < \sqrt{\gamma_1} + \sqrt{[(1-\alpha)/\alpha](\gamma_2 - \gamma_1)}$  and  $\gamma_2 / \sqrt{\gamma_1} < \sqrt{\gamma_2 / \alpha}$ . It follows that DRCC (4) is equivalent to  $b / \sqrt{y^\top \Sigma y} \geq \sqrt{\gamma_2 / \alpha}$ .

The proofs of the above two cases are both completed by the definition of  $b$ .  $\square$

To sum up, we derive exact 0-1 SOC program reformulations of the DCBP model under ambiguity sets  $\mathcal{D}_1$  and  $\mathcal{D}_2$ . We summarize these reformulations in the following proposition.

**Proposition 1.** The DCBP model with  $\mathcal{D} = \mathcal{D}_1$  is equivalent to the following 0-1 SOC program:

$$\min_{\mathbf{z}, \mathbf{y}} \left\{ \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} : (1b)-(1d), (5) \right\}. \quad (10a)$$

When  $\mathcal{D} = \mathcal{D}_2$  and  $\gamma_1 / \gamma_2 \leq \alpha$ , the DCBP model is equivalent to the following 0-1 SOC program:

$$\min_{\mathbf{z}, \mathbf{y}} \left\{ \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} : (1b)-(1d), (6a) \right\}. \quad (10b)$$

When  $\mathcal{D} = \mathcal{D}_2$  and  $\gamma_1 / \gamma_2 > \alpha$ , the DCBP model is equivalent to the following 0-1 SOC program:

$$\min_{\mathbf{z}, \mathbf{y}} \left\{ \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} : (1b)-(1d), (6b) \right\}. \quad (10c)$$

## 4 Valid Inequalities for DCBP

In Section 3, we derive three 0-1 SOC reformulations (10a)–(10c) of the DCBP model. Although these reformulations can be directly handled by the off-the-shelf solvers, as we report in Section 5, these 0-1 SOC programs are often time-consuming to solve, primarily because of the binary restrictions on variables  $y$  and  $z$ . In this section, we derive valid inequalities of the DCBP formulation, with the objective of accelerating the branch-and-cut algorithm for solving DCBP and also general 0-1 SOC programs in commercial solvers. To this end, we exploit (a) the submodularity of 0-1 SOC constraints and (b) the bin packing structure of DCBP. In Section 4.1, we derive a sufficient condition for submodularity and two approximations of the 0-1 SOC constraints that satisfy this condition. Using the submodular approximations, we derive extended polymatroid inequalities. In Section 4.2, we show the submodularity of the 0-1 SOC constraints in a lifted (i.e., higher-dimensional) space. Additionally, we derive valid inequalities in the lifted space based on the bin packing structure.

### 4.1 Submodularity of the 0-1 SOC Constraints

We consider SOC constraints of the form

$$\mu^\top y + \sqrt{y^\top \Lambda y} \leq T, \quad (11)$$

where  $\Lambda$  represents a  $J \times J$  positive definite matrix. Note that all SOC reformulations (5), (6a), and (6b) derived in Section 3, as well as the SOC reformulation (3) of a chance constraint with Gaussian uncertainty, possess the form of (11). Before investigating the submodularity of (11), we review the definitions of submodular functions and extended polymatroids.

**Definition 1.** Define set  $\mathcal{J} := \{1, \dots, J\}$  and the collection of its subsets  $\mathcal{C} := \{R : R \subseteq \mathcal{J}\}$ . A set function  $h: \mathcal{C} \rightarrow \mathbb{R}$  is called submodular if and only if  $h(R \cup \{j\}) - h(R) \geq h(S \cup \{j\}) - h(S)$  for all  $R \subseteq S \subseteq \mathcal{J}$  and all  $j \in \mathcal{J} \setminus S$ .

Throughout this section, we refer to a set function as  $h(R)$  and  $h(y)$  interchangeably, where  $y \in \{0, 1\}^J$  represents the indicating vector for subset  $R \subseteq \mathcal{J}$ , i.e.,  $y_j = 1$  if  $j \in R$  and  $y_j = 0$  otherwise.

**Definition 2.** For a submodular function  $h(y)$ , the polyhedron

$$\text{EP}_h = \{\pi \in \mathbb{R}^J : \pi(R) \leq h(R), \quad \forall R \subseteq \mathcal{J}\}$$

is called an extended polymatroid associated with  $h(y)$ , where  $\pi(R) = \sum_{j \in R} \pi_j$ .

For a submodular function  $h(y)$  with  $h(\emptyset) = 0$ , inequality

$$\pi^\top y \leq t, \quad (12)$$

termed an *extended polymatroid inequality*, is valid for the epigraph of  $h$ , i.e.,

$$\{(y, t) \in \{0, 1\}^J \times \mathbb{R} : t \geq h(y)\}, \text{ if and only if } \pi \in \text{EP}_h.$$

Furthermore, the separation of (12) can be efficiently done by a greedy algorithm (Edmonds, 1970), which we briefly describe in Algorithm 1.

---

**Algorithm 1** A greedy algorithm for separating extended polymatroid inequalities (12).

---

**Input:** A point  $(\hat{y}, \hat{t})$  with  $\hat{y} \in [0, 1]^J$ ,  $\hat{t} \in \mathbb{R}$ , and a submodular function  $h : \{0, 1\}^J \rightarrow \mathbb{R}$ .

- 1: Sort the entries in  $\hat{y}$  such that  $y_{(1)} \geq \dots \geq y_{(J)}$ . Obtain the permutation  $\{(1), \dots, (J)\}$  of  $\mathcal{J}$ .
  - 2: Letting  $R_{(j)} := \{(1), \dots, (j)\}$ ,  $\forall j \in \mathcal{J}$ , compute  $\hat{\pi}_{(1)} = h(R_{(1)})$ , and  $\hat{\pi}_{(j)} = h(R_{(j)}) - h(R_{(j-1)})$  for  $j = 2, \dots, J$ .
  - 3: **if**  $\hat{\pi}^\top \hat{y} > \hat{t}$  **then**
  - 4:   We generate a valid extended polymatroid inequality  $\hat{\pi}^\top y \leq t$ .
  - 5: **else**
  - 6:   The current solution  $(\hat{y}, \hat{t})$  satisfies  $h(\hat{y}) \leq \hat{t}$ .
  - 7: **end if**
  - 8: **return** either  $(\hat{y}, \hat{t})$  is feasible, or a violated extended polymatroid inequality  $\hat{\pi}^\top y \leq t$ .
- 

The strength and efficient separation of the extended polymatroid inequality motivate us to explore the submodularity of function  $g(y) := \mu^\top y + \sqrt{y^\top \Lambda y}$ . In a special case,  $\Lambda$  is assumed to be a diagonal matrix and so the item weights  $\tilde{t}_{ij}$  in the same bin are uncorrelated. In this case, Atamtürk and Narayanan (2008) successfully show that  $g(y)$  is submodular. As a result, we can strengthen the DCBP formulation by incorporating extended polymatroid inequalities in the form  $\pi^\top y \leq T$ , where  $\pi \in \text{EP}_g$ . Unfortunately, the submodularity of  $g(y)$  quickly fades when the off-diagonal entries of  $\Lambda$  become non-zero, e.g., when  $\Lambda$  is associated with a general covariance matrix. We present an example as follows.

**Example 1.** Suppose that  $\mathcal{J} = \{1, 2, 3\}$ ,  $\mu = 0$ , and

$$\Lambda = \begin{bmatrix} 0.6 & -0.2 & 0.2 \\ -0.2 & 0.7 & 0.1 \\ 0.2 & 0.1 & 0.6 \end{bmatrix}.$$

The three eigenvalues of  $\Lambda$  are 0.2881, 0.7432, and 0.8687, and so  $\Lambda \succ 0$ . However, function  $g(y) = \mu^\top y + \sqrt{y^\top \Lambda y}$  is not submodular because  $h(R \cup \{j\}) - h(R) < h(S \cup \{j\}) - h(S)$ , where  $R = \{1\}$ ,  $S = \{1, 2\}$ , and  $j = 3$ .  $\square$

In this section, we provide a sufficient condition for function  $g(y)$  being submodular for general  $\Lambda$ . To this end, we derive a necessary and sufficient condition for quadratic function  $y^\top \Lambda y$  being submodular in the following theorem.

**Theorem 3.** Define function  $h : \{0, 1\}^J \rightarrow \mathbb{R}$  such that  $h(y) := y^\top \Lambda y$ , where  $\Lambda \in \mathbb{R}^{J \times J}$  represents a symmetric matrix. Then,  $h(y)$  is submodular if and only if  $\Lambda_{rs} \leq 0$  for all  $r, s = 1, \dots, J$  and  $r \neq s$ .

*Proof.* **(If)** Pick any  $R, S \subseteq \mathcal{J}$  and  $k \in \mathcal{J}$  such that  $R \subseteq S$  and  $k \in \mathcal{J} \setminus S$ . Then,  $h(R) = e_R^\top \Lambda e_R$ , where  $e_R \in \{0, 1\}^J$  is such that  $[e_R]_j = 1$  if  $j \in R$  and  $[e_R]_j = 0$  otherwise. Similarly, we have  $h(S) = e_S^\top \Lambda e_S$ ,  $h(R \cup \{k\}) = (e_R + e_k)^\top \Lambda (e_R + e_k) = e_R^\top \Lambda e_R + 2e_R^\top \Lambda e_k + e_k^\top \Lambda e_k$ , and  $h(S \cup \{k\}) = (e_S + e_k)^\top \Lambda (e_S + e_k) = e_S^\top \Lambda e_S + 2e_S^\top \Lambda e_k + e_k^\top \Lambda e_k$ . It follows that

$$\begin{aligned} \left[ h(R \cup \{k\}) - h(R) \right] - \left[ h(S \cup \{k\}) - h(S) \right] &= \left( 2e_R^\top \Lambda e_k + e_k^\top \Lambda e_k \right) - \left( 2e_S^\top \Lambda e_k + e_k^\top \Lambda e_k \right) \\ &= 2 \sum_{j \in R} \Lambda_{jk} - 2 \sum_{j \in S} \Lambda_{jk} \\ &= -2 \sum_{j \in S \setminus R} \Lambda_{jk} \geq 0, \end{aligned}$$

where the second equality is due to the definition of vectors  $e_R$  and  $e_S$ , and the last inequality is due to  $\Lambda_{jk} \leq 0$  because  $j \in S \setminus R$  and  $k \in \mathcal{J} \setminus S$ , and so  $j \neq k$ . Hence,  $h(R \cup \{k\}) - h(R) \geq h(S \cup \{k\}) - h(S)$  and so function  $h(y)$  is submodular.

**(Only If)** We prove by contradiction. Suppose that function  $h(y)$  is submodular and there exist  $r, s \in \mathcal{J}$  such that  $r \neq s$  and  $\Lambda_{rs} > 0$ . Let  $R = \emptyset$ ,  $S = \{s\}$ . Then,  $h(R) = 0$ ,  $h(S) = \Lambda_{ss}$ ,  $h(R \cup \{r\}) = \Lambda_{rr}$ , and  $h(S \cup \{r\}) = \Lambda_{rr} + \Lambda_{ss} + 2\Lambda_{rs}$ . It follows that

$$\left[ h(R \cup \{r\}) - h(R) \right] - \left[ h(S \cup \{r\}) - h(S) \right] = -2\Lambda_{rs} < 0,$$

which contradicts that function  $h(y)$  is submodular.  $\square$

Note that Theorem 3 does not assume  $\Lambda \succeq 0$  and so can be applied to general (convex or non-convex) quadratic functions. Theorem 3 leads to a sufficient condition for function  $g(y)$  being submodular. We summarize this result in the following proposition.

**Proposition 2.** Let  $\Lambda \in \mathbb{R}^{J \times J}$  represent a symmetric and positive semidefinite matrix that satisfies (i)  $2 \sum_{s=1}^J \Lambda_{rs} \geq \Lambda_{rr}$  for all  $r = 1, \dots, J$  and (ii)  $\Lambda_{rs} \leq 0$  for all  $r, s = 1, \dots, J$  and  $r \neq s$ . Then, function  $g(y) = \mu^\top y + \sqrt{y^\top \Lambda y}$  is submodular.

*Proof.* As  $\mu^\top y$  is submodular in  $y$ , it suffices to prove that  $\sqrt{y^\top \Lambda y}$  is submodular. Hence, we can assume  $\mu = 0$  without loss of generality. We let  $f(x) = \sqrt{x}$  and  $h(y) = y^\top \Lambda y$ . Then,  $g(y) = f(h(y))$ .

First, we note that  $h(y \vee e_r) \geq h(y)$  for all  $y \in \{0, 1\}^J$  and  $r = 1, \dots, J$ , where  $a \vee b = [\max\{a_1, b_1\}, \dots, \max\{a_J, b_J\}]^\top$  for  $a, b \in \mathbb{R}^J$ . Indeed, if  $y_r = 1$  then  $y \vee e_r = y$  and so  $h(y \vee e_r) = h(y)$ . Otherwise, if  $y_r = 0$ , then  $y \vee e_r = y + e_r$  and so

$$\begin{aligned} h(y \vee e_r) &= y^\top \Lambda y + 2e_r^\top \Lambda y + e_r^\top \Lambda e_r \\ &= y^\top \Lambda y + 2 \sum_{s: y_s=1} \Lambda_{rs} + \Lambda_{rr} \\ &\geq y^\top \Lambda y + 2 \sum_{\substack{s=1 \\ s \neq r}}^J \Lambda_{rs} + \Lambda_{rr} \\ &\geq y^\top \Lambda y = h(y), \end{aligned}$$

where the first inequality is due to  $y_r = 0$  and condition (ii), and the second inequality is due to condition (i). It follows that  $h(y') \geq h(y)$  for all  $y, y' \in \{0, 1\}^J$  such that  $y' \geq y$ . Hence,  $h(y)$  is increasing.

Second, based on Theorem 3,  $h(y)$  is submodular due to condition (ii). It follows that  $g(y) = f(h(y))$  is submodular because function  $f$  is concave and nondecreasing and function  $h$  is submodular and increasing (see, e.g., Proposition 2.2.5, Simchi-Levi et al., 2014)<sup>1</sup>.  $\square$

Proposition 2 generalizes the sufficient condition in Atamtürk and Narayanan (2008) because conditions (i)–(ii) are automatically satisfied if  $\Lambda$  is diagonal and positive definite. For general  $\Lambda \succ 0$  that does not satisfy sufficient conditions (i)–(ii), we can approximate SOC constraint (11) by replacing  $\Lambda$  with a matrix  $\Delta$  that satisfies these conditions. We derive relaxed and conservative submodular approximations of constraint (11) in the following proposition.

**Proposition 3.** Constraint (11) implies the SOC constraint

$$\mu^\top y + \sqrt{y^\top \Delta^L y} \leq T, \quad (13)$$

where function  $g^L(y) := \mu^\top y + \sqrt{y^\top \Delta^L y}$  is submodular and  $\Delta^L$  is an optimal solution of SDP

$$\min_{\Delta} \|\Delta - \Lambda\|_2 \quad (14a)$$

$$\text{s.t. } 0 \preceq \Delta \preceq \Lambda, \quad (14b)$$

$$2 \sum_{s=1}^J \Delta_{rs} \geq \Delta_{rr}, \quad \forall r = 1, \dots, J, \quad (14c)$$

$$\Delta_{rs} \leq 0, \quad \forall r, s = 1, \dots, J \text{ and } r \neq s. \quad (14d)$$

Additionally, constraint (11) is implied by the SOC constraint

$$\mu^\top y + \sqrt{y^\top \Delta^U y} \leq T, \quad (15)$$

where function  $g^U(y) := \mu^\top y + \sqrt{y^\top \Delta^U y}$  is submodular and  $\Delta^U$  is an optimal solution of SDP

$$\min_{\Delta} \|\Delta - \Lambda\|_2 \quad (16a)$$

$$\text{s.t. } \Delta \succeq \Lambda, \quad (14c)–(14d). \quad (16b)$$

*Proof.* By construction,  $g^L(y)$  is submodular because  $\Delta^L$  satisfies constraints (14c)–(14d) and so conditions (i)–(ii). Additionally, constraint (11) implies (13) because  $\Delta^L$  satisfies constraint (14b) and so  $\Delta^L \preceq \Lambda$ . Similarly, we obtain that  $g^U(y)$  is submodular and constraint (11) is implied by (15).  $\square$

**Remark 1.** There exist matrices  $\Delta^L$  and  $\Delta^U$  that are feasible to SDPs (14a)–(14d) and (16a)–(16b),

---

<sup>1</sup>Proposition 2.2.5 in Simchi-Levi et al. (2014) assumes that  $y \in \mathbb{R}^n$  and provides a sufficient condition for  $g$  being supermodular. It can be shown that a similar proof of this proposition applies to our case.

respectively. For example,  $\text{diag}(\lambda_{\min}, \dots, \lambda_{\min}) \in \mathbb{R}^{J \times J}$  satisfy constraints (14b)–(14d), where  $\lambda_{\min}$  represents the smallest eigenvalue of matrix  $\Lambda$ . Additionally,  $\text{diag}(\lambda_{\max}, \dots, \lambda_{\max}) \in \mathbb{R}^{J \times J}$  satisfy constraints (16b), where  $\lambda_{\max}$  represents the largest eigenvalue of matrix  $\Lambda$ .

**Remark 2.** Proposition 3 holds for general 0-1 SOC constraints. We can apply the relaxed and conservative submodular approximations (13) and (15) on any 0-1 SOC programs, e.g., the knapsack problem with DRCCs.

Note that by minimizing the  $\ell^2$  distance between  $\Delta$  and  $\Lambda$  in objective functions (14a) and (16a), we find “optimal” approximations of  $\Lambda$  that satisfies sufficient conditions (i)–(ii) in Proposition 2. Accordingly, we obtain “optimal” submodular approximations of the 0-1 SOC constraint (11). There are many possible alternatives of the  $\ell^2$  norm in (14a) and (16a). For example, formulations (14a)–(14d) and (16a)–(16b) remain SDPs if the  $\ell^2$  norm is replaced by the  $\ell^1$  norm or the  $\ell^\infty$  norm. In computation, we only need to solve these two SDPs once to obtain  $\Delta^L$  and  $\Delta^U$ . Then, extended polymatroid inequalities can be obtained from the relaxed approximation (13). Additionally, the conservative approximation (15) leads to an upper bound of the optimal objective value of DCBP. We summarize these observations in the following proposition.

**Proposition 4.** The optimal objective value of formulation

$$\min_{\mathbf{z}, \mathbf{y}} \left\{ \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} : (1b)–(1d), (13) \right\} \quad (17)$$

is less than or equal to that of the DCBP formulation with SOC constraint (11). Extended polymatroid inequalities  $\pi^\top y \leq T$  are valid for the DCBP formulation for all  $\pi \in \text{EP}_{g^L}$ . Additionally, the optimal objective value of formulation

$$\min_{\mathbf{z}, \mathbf{y}} \left\{ \sum_{i=1}^I c_i^z z_i + \sum_{i=1}^I \sum_{j=1}^J c_{ij}^y y_{ij} : (1b)–(1d), (15) \right\} \quad (18)$$

is greater than or equal to that of the DCBP formulation with SOC constraint (11). Extended polymatroid inequalities  $\pi^\top y \leq T$  are valid for (18) for all  $\pi \in \text{EP}_{g^U}$ .

## 4.2 Valid Inequalities in a Lifted Space

In Section 4.1, we derive extended polymatroid inequalities for DCBP based on a relaxed approximation of SOC constraint (11). These valid inequalities might not be facet-defining if matrix  $\Lambda$  does not satisfy sufficient conditions (i)–(ii) in Proposition 2. In this section, we show that the submodularity of (11) holds for general  $\Lambda$  in a lifted (i.e., higher-dimensional) space. Accordingly, we derive extended polymatroid inequalities in the lifted space.

To this end, we reformulate constraint (11) as  $\mu^\top y \leq T$  and  $y^\top \Lambda y \leq (T - \mu^\top y)^2$ , i.e.,  $y^\top (\Lambda - \mu\mu^\top) y + 2T\mu^\top y \leq T^2$ . Then, we define  $w_{jk} = y_j y_k$  for all  $j, k = 1, \dots, J$  and augment vector  $y$  to vector  $v = [y_1, \dots, y_J, w_{11}, \dots, w_{1J}, w_{21}, \dots, w_{JJ}]^\top$ . We can incorporate the following McCormick



inequalities to define each  $w_{ij}$ :

$$w_{jk} \leq y_j, \quad w_{jk} \leq y_k, \quad w_{jk} \geq y_j + y_k - 1, \quad w_{ij} \geq 0. \quad (19)$$

Accordingly, we rewrite (11) as  $[\mu^\top, 0^\top]v \leq T$  and

$$a^\top v + v^\top B_N v \leq T^2, \quad (20)$$

where we decompose  $(\Lambda - \mu\mu^\top)$  to be the sum of two matrices, one containing all positive entries and the other containing all nonpositive entries. Accordingly, we define vector  $a := [2T\mu; B_P]^\top$  with  $B_P \in \mathbb{R}_+^{J^2}$  representing all the positive entries after vectorization, and matrix  $B_N \in \mathbb{R}_-^{(J+J^2) \times (J+J^2)}$  collects all nonpositive entries, i.e.,  $B_N = \begin{bmatrix} 0 & 0 \\ 0 & -(\Lambda - \mu\mu^\top)_- \end{bmatrix}$ , where  $(x)_- = -\min\{0, x\}$  for  $x \in \mathbb{R}$ .

As  $a^\top v + v^\top B_N v$  is a submodular function of  $v$  by Theorem 3, we can incorporate extended polymatroid inequalities to strengthen the lifted SOC constraints (11). We summarize this result in the following proposition.

**Proposition 5.** Define function  $h : \{0, 1\}^{J+J^2} \rightarrow \mathbb{R}$  such that  $h(v) := a^\top v + v^\top B_N v$ . Then,  $h$  is submodular. Furthermore, inequality  $\pi^\top v \leq T^2$  is valid for set  $\{v \in \{0, 1\}^{J+J^2} : h(v) \leq T^2\}$  for all  $\pi \in \text{EP}_h$  and the separation of this inequality can be done by Algorithm 1.

Note that this lifting procedure introduces  $J^2$  additional variables  $w_{ijk}$  for each bin  $i$ . Although  $w_{ijk}$  can be treated as continuous variables when solving DCBP in view of the McCormick inequalities (19), we have by far omitted the bin packing structure (e.g., constraints (1b)–(1d)) of the DCBP model. Here, we derive valid inequalities in the lifted space containing variables  $z_i$ ,  $y_{ij}$ , and  $w_{ijk}$  to further strengthen the DCBP formulation. We summarize the main results in the following two propositions.

**Proposition 6.** For all extended polymatroid inequalities  $\pi^\top y_i \leq T$  with regard to bin  $i$ ,  $\forall i = 1, \dots, I$ , inequality

$$\pi^\top y_i \leq T z_i \quad (21a)$$

is valid for the DCBP formulation. Similarly, for all extended polymatroid inequalities  $\pi^\top v_i \leq T^2$  with regard to bin  $i$ ,  $\forall i = 1, \dots, I$ , inequality

$$\pi^\top v_i \leq T^2 z_i \quad (21b)$$

is valid for the DCBP formulation.

*Proof.* When  $z_i = 1$ , inequality (21a) reduces to the extended polymatroid inequality. When  $z_i = 0$ , we have  $y_{ij} = 0$  for all  $j = 1, \dots, J$  due to constraints (1b). It follows that inequality (21a) holds valid.

When  $z_i = 1$ , inequality (21b) reduces to the extended polymatroid inequality. When  $z_i = 0$ , we have  $y_{ij} = 0$  for all  $j = 1, \dots, J$  due to constraints (1b) and so  $w_{ijk} = 0$  for all  $j, k = 1, \dots, J$ .

It follows that  $v_i = 0$  by definition. Hence, inequality (21b) holds valid.  $\square$

**Proposition 7.** Consider set

$$L = \left\{ (z, y, w) \in \{0, 1\}^{I \times (IJ) \times (IJ^2)} : (1b)-(1d), w_{ijk} = y_{ij}y_{ik}, \forall j, k = 1, \dots, J \right\}.$$

Without loss of optimality, the following inequalities are valid for  $L$ :

$$w_{ijk} \geq y_{ij} + y_{ik} + \sum_{\substack{\ell=1 \\ \ell \neq i}}^I w_{\ell jk} - 1 \quad \forall j, k = 1, \dots, J \quad (22a)$$

$$w_{ijk} \geq y_{ij} + y_{ik} - z_i \quad \forall i = 1, \dots, I, \forall j, k = 1, \dots, J \quad (22b)$$

$$\sum_{\substack{j=1 \\ j \neq k}}^J w_{ijk} \leq \sum_{j=1}^J y_{ij} - z_i \quad \forall i = 1, \dots, I, \forall k = 1, \dots, J \quad (22c)$$

$$\sum_{j=1}^J \sum_{\substack{k=1 \\ k \neq j}}^J w_{ijk} \geq \sum_{j=1}^J y_{ij} - z_i \quad \forall i = 1, \dots, I. \quad (22d)$$

*Proof.* (**Validity of inequality (22a)**) If  $j = k$ , then  $w_{ijk} = y_{ij}^2 = y_{ij}$ . In this case, inequality (22a) reduces to  $y_{ij} \geq 2y_{ij} + \sum_{\ell=1, \ell \neq i}^I y_{\ell j} - 1$ , which clearly holds because  $y_{ij} + \sum_{\ell=1, \ell \neq i}^I y_{\ell j} = \sum_{i=1}^I y_{ij} \leq 1$ . If  $j \neq k$ , then we discuss the following two cases:

1. If  $\max\{y_{ij}, y_{ik}\} = 1$ , then we assume  $y_{ij} = 1$  without loss of generality. It follows that  $y_{\ell j} = 0$  due to constraints (1c) and so  $w_{\ell jk} = y_{\ell j}y_{\ell k} = 0$  for all  $\ell \neq i$ . Hence,  $\sum_{\ell=1, \ell \neq i}^I w_{\ell jk} = 0$  and inequality (22a) reduces to  $w_{ijk} \geq y_{ij} + y_{ik} - 1$ , which holds valid.
2. If  $\max\{y_{ij}, y_{ik}\} = 0$ , then  $y_{ij} = y_{ik} = 0$  and  $w_{ijk} = 0$ . It remains to show  $\sum_{\ell=1, \ell \neq i}^I w_{\ell jk} \leq 1$ . Indeed, since  $w_{\ell jk} \leq y_{\ell j}$ , we have  $\sum_{\ell=1, \ell \neq i}^I w_{\ell jk} \leq \sum_{\ell=1, \ell \neq i}^I y_{\ell j} \leq \sum_{\ell=1}^I y_{\ell j} = 1$ , where the last equality is due to constraints (1c).

(**Validity of inequality (22b)**) This inequality clearly holds valid when  $z_i = 1$ . When  $z_i = 0$ , we have  $y_{ij} = y_{ik} = 0$  due to constraints (1b). It follows that  $w_{ijk} = y_{ij}y_{ik} = 0$  and so the inequality holds valid.

(**Validity of inequality (22c)**) This inequality holds valid when  $z_i = 0$ . When  $z_i = 1$ , this inequality is equivalent to  $y_{ik} \sum_{\substack{j=1 \\ j \neq k}}^J y_{ij} \leq \sum_{j=1, j \neq k}^J y_{ij} - 1$ . We discuss the following two cases:

1. If  $y_{ik} = 0$ , then  $\sum_{j=1}^J y_{ij} \geq 1$  without loss of optimality because  $z_i = 1$ . Inequality (22c) holds valid.
2. If  $y_{ik} = 1$ , then  $y_{ik} \sum_{\substack{j=1 \\ j \neq k}}^J y_{ij} = \sum_{\substack{j=1 \\ j \neq k}}^J y_{ij}$ . Meanwhile,  $\sum_{j=1}^J y_{ij} - 1 = \sum_{\substack{j=1 \\ j \neq k}}^J y_{ij} + y_{ik} - 1 = \sum_{\substack{j=1 \\ j \neq k}}^J y_{ij}$ . Inequality (22c) holds valid.

(**Validity of inequality (22d)**) This inequality holds valid when  $z_i = 0$ . When  $z_i = 1$ , we have  $\sum_{j=1}^J y_{ij} \geq 1$  without loss of optimality. It follows that  $\sum_{j=1}^J y_{ij} = 1$  or  $\sum_{j=1}^J y_{ij} \geq 2$ , and so  $(\sum_{j=1}^J y_{ij} - 1)(\sum_{j=1}^J y_{ij} - 2) \geq 0$ . Hence,

$$\begin{aligned}
\left(\sum_{j=1}^J y_{ij} - 1\right)\left(\sum_{j=1}^J y_{ij} - 2\right) &= \sum_{j=1}^J y_{ij}^2 + 2 \sum_{j=1}^J \sum_{\substack{k=1 \\ k \neq j}}^J y_{ij} y_{ik} - 3 \left(\sum_{j=1}^J y_{ij}\right) + 2 \\
&= \sum_{j=1}^J y_{ij} + 2 \sum_{j=1}^J \sum_{\substack{k=1 \\ k \neq j}}^J w_{ijk} - 3 \left(\sum_{j=1}^J y_{ij}\right) + 2 \\
&= 2 \sum_{j=1}^J \sum_{\substack{k=1 \\ k \neq j}}^J w_{ijk} - 2 \left[\left(\sum_{j=1}^J y_{ij}\right) - 1\right] \geq 0.
\end{aligned}$$

Inequality (22d) follows.  $\square$

**Remark 3.** We note that valid inequalities (22a)–(22d) are *polynomially many* and all coefficients are in *closed-form*. Hence, we do not need any separation processes for these inequalities, and we can incorporate them in the DCBP formulation without dramatically increasing its size.

## 5 Numerical Studies

In this section, we numerically evaluate the performance of our proposed models and solution approaches. Our results consist of two parts, which report the CPU time (Section 5.2) and the out-of-sample performance of various methods (Section 5.3), respectively. More specifically, Section 5.2 demonstrates the computational efficacy of the valid inequalities we derived in Section 4 for the original or lifted SOC constraints. Section 5.3 demonstrates that DCBP solutions can well protect against the distributional ambiguity as opposed to the solutions obtained by following the Gaussian distribution assumption or by the SAA method. The DCBP solutions have very low probabilities of exceeding the capacities of open bins in all the out-of-sample tests, even when the distributional information is misspecified.

### 5.1 Computational Setup

We study DCBP in the context of server allocation for completing a set of appointments, each having random service time on different servers. We first consider  $I = 6$  servers (i.e., bins) and  $J = 32$  appointments (i.e., items) to test the DCBP model under various distributional assumptions and ambiguity sets. The daily operating time limit (i.e., capacity)  $T_i$  of each server  $i$  varies in between  $[420, 540]$  minutes (i.e., 7–9 hours). We let the opening cost  $c_i^z$  of each server  $i$  be an increasing function of  $T_i$  such that  $c_i^z = T_i^2/3600 + 3T_i/60$ , and let all assignment costs  $c_{ij}^y$ ,  $\forall i = 1, \dots, I$ ,  $j = 1, \dots, J$  vary in between  $[0, 18]$ , so that the total opening cost and the total assignment cost have similar magnitudes.

To generate samples of random service time (i.e., random item weight), we consider “high mean (hM)” and “low mean ( $\ell$ M)” being 25 minutes and 12.5 minutes, respectively. We set the coefficient of variation (i.e., the ratio of the standard deviation to the mean) as 1.0 for the “high variance (hV)” case and as 0.3 for the “low variance ( $\ell$ V)” case. We equally mix all four types of appointments with “hMhV”, “hM $\ell$ V”, “ $\ell$ MhV”, “ $\ell$ M $\ell$ V”, and thus have eight appointments of each type. We sample 10,000 data points as the random service time of each appointment on each server, following a Gaussian distribution with the above settings of mean and standard deviation. We will hereafter call them the in-sample data. To formulate the 0-1 SOC models with diagonal covariance matrices, we use the empirical mean and standard deviation of each  $\tilde{t}_{ij}$  obtained from the in-sample data and set  $\alpha_i = 0.05$ ,  $\forall i = 1, \dots, I$ . Using the same  $\alpha_i$ -values, we formulate the 0-1 SOC models under general covariance matrices, for which we use the empirical mean and covariance matrix obtained from the in-sample data. The empirical covariance matrices we obtain have most of their off-diagonal entries being non-zero, and some being quite significant.

All the computation is performed on a Windows 7 machine with Intel(R) Core(TM) i7-2600 CPU 3.40 GHz and 8GB memory. We implement the optimization models and the branch-and-cut algorithm using commercial solver GUROBI 5.6.3 via Python 2.7.10. The GUROBI default settings are used for optimizing all the 0-1 programs, and we set the number of threads as one. When implementing the branch-and-cut algorithm, we add the violated extended polymatroid inequalities using GUROBI `callback` class by `Model.cbCut()` for both integer and fractional temporary solutions. For all the nodes in the branch-and-bound tree, we generate violated cuts at each node as long as any exists. The optimality gap tolerance is set as 0.01%. We also set the threshold for identifying violated cuts as  $10^{-4}$ , and set the time limit for computing each instance as 3600 seconds.

## 5.2 CPU Time Comparison

Here, we solve 0-1 SOC reformulations or approximations of DCBP, and use either a diagonal or a general covariance matrix in each model. Our valid inequalities significantly reduce the solution time of directly solving the 0-1 SOC models in GUROBI, while the extended polymatroid inequalities generated based on the approximate and lifted SOC constraints perform differently depending on the problem size. The details are presented as follows.

### 5.2.1 Computing 0-1 SOC models with diagonal matrices

We first optimize 0-1 SOC models with a diagonal matrix in constraint (11), of which the left-hand side function  $g(y)$  is submodular, and thus we use extended polymatroid inequalities (21a) with  $\pi \in EP_g$  in a branch-and-cut algorithm. Table 1 presents the CPU time (in seconds), optimal objective values, and other solution details (including “**Server**” as the number of open servers, “**Node**” as the total number of branching nodes, and “**Cut**” as the total number of cuts added) for solving the three 0-1 SOC models **DCBP1** (i.e., (10a) using ambiguity set  $\mathcal{D}_1$ ), **DCBP2** (i.e., (10b) or (10c) using ambiguity set  $\mathcal{D}_2$  with  $\gamma_1 = 1$ ,  $\gamma_2 = 2$ ), and **Gaussian** (assuming Gaussian

distributed service time). We also implement the SAA approach (i.e., row “**SAA**”) by optimizing the MILP reformulation of the chance-constrained bin packing model based on the 10,000 in-sample data points. We compare the branch-and-cut algorithm using our extended polymatroid inequalities (in rows “**B&C**”) with directly solving the 0-1 SOC models in GUROBI (in rows “**w/o Cuts**”).

Table 1: CPU time and solution details for solving instances with diagonal matrices

Approach	Model	Time (s)	Opt. Obj.	Server	Opt. Gap	Node	Cut
B&C	DCBP1	0.73	328.99	3	0.00%	83	82
	DCBP2	27.50	366.54	3	0.00%	2146	2624
	Gaussian	0.13	297.94	2	0.00%	0	0
w/o Cuts	DCBP1	95.73	328.99	3	0.01%	76237	N/A
	DCBP2	LIMIT	380.09	2	9.15%	409422	N/A
	Gaussian	0.02	297.94	2	0.00%	16	N/A
SAA	MILP	21.20	297.94	2	0.00%	89	N/A

In Table 1, the branch-and-cut algorithm quickly optimizes DCBP1 and DCBP2. Especially, if being directly solved by GUROBI, DCBP2 cannot be solved within the 3600-second time limit and ends with a 9.15% optimality gap. Solving DCBP1 by using the branch-and-cut algorithm is much faster than solving the large-scale SAA-based MILP model, while the solution time of DCBP2 is similar to the latter. The two DCBP models also yield higher objective values, since they both provide more conservative solutions that open one more server than either the Gaussian or the SAA-based approach.

### 5.2.2 Computing 0-1 SOC models with general covariance matrices

In this section, we focus on testing DCBP2 yielded by the ambiguity set  $\mathcal{D}_2$  with parameters  $\gamma_1 = 1$ ,  $\gamma_2 = 2$ , and  $\alpha_i = 0.05$ ,  $\forall i = 1, \dots, I$ . We use empirical covariance matrices of the in-sample data. Note that these covariance matrices are general and non-diagonal. We compare the time of solving the 0-1 SOC reformulations of DCBP2 on ten independently generated instances. We examine two implementations of the branch-and-cut algorithm: one uses extended polymatroid inequalities (21a) with  $\pi \in EP_{g^L}$  based on the relaxed 0-1 SOC constraint (13), and the other uses the extended polymatroid inequalities (21b) based on the lifted SOC constraint.

Table 2 reports the CPU time (in seconds) and number of branching nodes (in column “**Node**”) for various methods. First, we directly solve the 0-1 SOC models of DCPB2 in GUROBI, without or with the linear valid inequalities (22a)–(22d), and report their results in columns “**w/o Cuts**” and “**Ineq.**”, respectively. We then implement the branch-and-cut algorithm, and examine the results of using extended polymatroid inequalities (21a) with  $\pi \in EP_{g^L}$  (reported in columns “**B&C-Relax**”) and cuts (21b) based on lifted SOC constraints (reported in columns “**B&C-Lifted**”). For both B&C methods, we also present the number of extended polymatroid inequalities (see column “**Cut**”) added.

In Table 2, we highlight the solution time of the method that runs the fastest for each instance. Note that without the extended polymatroid inequalities or the valid inequalities, the GUROBI

Table 2: CPU time of DCBP2 solved by different methods with general covariance matrices

Instance	w/o Cuts		Ineq.		B&C-Relax			B&C-Lifted		
	Time (s)	Node	Time (s)	Node	Time (s)	Node	Cut	Time (s)	Node	Cut
1	286.29	10409	156.50	795	51.99	9095	702	<b>35.03</b>	618	823
2	433.32	10336	167.91	687	26.63	6524	698	<b>12.34</b>	405	235
3	284.17	10434	206.82	971	70.43	17420	621	<b>29.84</b>	595	729
4	310.11	10302	139.06	656	<b>15.37</b>	2467	723	25.31	419	617
5	329.32	10453	181.83	777	56.53	12349	737	<b>35.09</b>	678	921
6	365.28	10300	168.26	652	<b>23.89</b>	4807	695	26.73	555	595
7	296.55	10759	198.87	873	45.21	11585	738	<b>21.08</b>	440	626
8	278.62	10490	211.05	900	53.84	14540	721	<b>47.78</b>	1064	1686
9	139.24	7771	177.41	632	19.90	3918	645	<b>19.37</b>	216	360
10	297.72	10330	159.52	822	30.36	6877	649	<b>29.43</b>	400	727

solver takes the longest time for solving all the instances except instance #9. Adding the valid inequalities (22a)–(22d) to the solver reduces the solution time by 40% or more in almost all the instances, and drastically reduces the number of branching nodes. The extended polymatroid inequalities (21a)–(21b) further reduce the CPU time significantly (see columns “**B&C-Relax**” and “**B&C-Lifted**”). Moreover, for all the instances having 6 servers and 32 appointments, the algorithm using the extended polymatroid inequalities (21b) runs faster in eight out of ten instances than the algorithm using cuts (21a) based on relaxed SOC constraints without lifting. It indicates that the extended polymatroid inequalities generated by the lifted SOC constraints are more effective than those generated by the relaxed SOC constraints. This observation is overturned when we later increase the problem size.

In the following, we continue reporting the CPU time of solving DCBP2 with general covariance matrix. We vary the problem sizes (i.e., values of  $I$  and  $J$ ) in Section 5.2.3, and vary the values of  $\Lambda$  in the SOC constraint (11) in Section 5.2.4.

### 5.2.3 Solving 0-1 SOC models under different problem sizes

We use the same problem settings as in Section 5.1, and vary  $I = 6, 8, 10$  and  $J = 32, 40$  to test DCBP2 instances with different sizes. We still keep an equal mixture of all the four appointment types in each instance. Table 3 presents the computational time (in seconds), the total number of branching nodes (“**Node**”), and the total number of extended polymatroid inequalities generated (“**Cuts**”; if applicable) for solving the 0-1 SOC reformulation of DCBP2 by directly using GUROBI (“**w/o Cuts**”) and by using the two implementations of the extended polymatroid inequalities (“**B&C-Relax**” and “**B&C-Lifted**”).

In Table 3, we again highlight the solution time of the method that runs the fastest in each instance. We keep the first five instances we reported in Table 2 for instances with  $I = 6$ ,  $J = 32$ , and report five instances for other  $(I, J)$  combinations. From Table 3, we observe that both implementations of the extended polymatroid inequalities run significantly faster than directly using GUROBI, especially when we increase the problem sizes (i.e.,  $I$  increased from 6 to 10, and

Table 3: CPU time of DCBP2 with general covariance matrices for different problem sizes

Method	Inst.	$J = 32$					$J = 40$						
		1	2	3	4	5	6	7	8	9	10		
$I = 6$	B&C-Relax	Time (s)	51.99	26.63	70.43	<b>15.37</b>	56.53	<b>6.87</b>	<b>12.76</b>	<b>1.59</b>	<b>2.36</b>	<b>12.73</b>	
		Node	9095	6524	17420	2467	12349	1009	1322	176	285	1270	
		Cut	702	698	621	723	737	174	604	171	179	602	
	B&C-Lifted	Time (s)	<b>35.03</b>	<b>12.34</b>	<b>29.84</b>	25.31	<b>35.09</b>	64.58	98.18	91.12	60.11	59.50	
		Node	618	405	595	419	678	274	484	447	289	234	
		Cut	823	235	729	617	921	470	690	688	462	394	
	w/o Cuts	Time (s)	286.29	433.32	284.17	310.11	329.32	1654.31	208.12	1182.46	1580.41	1266.27	
		Node	10409	10336	10434	10302	10453	10525	1272	10732	10658	10642	
	$I = 8$	B&C-Relax	Time (s)	<b>41.57</b>	139.41	<b>55.22</b>	261.24	305.72	<b>23.91</b>	<b>9.73</b>	<b>17.76</b>	<b>27.16</b>	<b>12.98</b>
			Node	8342	29042	12267	49820	61334	2130	1240	1561	2607	1024
Cut			737	770	742	803	790	714	199	728	702	690	
B&C-Lifted		Time (s)	106.03	<b>28.55</b>	84.64	<b>97.05</b>	<b>13.56</b>	331.29	273.14	307.06	178.41	161.39	
		Node	678	502	647	634	125	1177	836	1397	457	529	
		Cut	114	691	128	143	216	1781	1175	2066	703	719	
w/o Cuts		Time (s)	866.12	597.43	649.72	683.18	497.15	2265.53	2428.60	2294.62	1781.95	851.99	
		Node	10338	10305	10309	10306	14386	11441	11219	11708	11128	5241	
$I = 10$		B&C-Relax	Time (s)	<b>3.75</b>	<b>9.28</b>	<b>6.56</b>	<b>3.23</b>	<b>16.71</b>	<b>29.94</b>	<b>80.34</b>	<b>22.58</b>	<b>24.48</b>	<b>339.93</b>
			Node	637	972	659	549	2274	2336	7315	1870	1959	34306
	Cut		241	552	390	230	741	767	714	736	729	715	
	B&C-Lifted	Time (s)	108.43	117.44	120.60	22.10	111.37	186.72	714.45	197.42	549.90	661.13	
		Node	668	785	828	291	779	766	1108	811	896	1209	
		Cut	108	191	314	281	188	1196	850	1106	568	808	
	w/o Cuts	Time (s)	987.92	1140.23	183.06	1113.09	1425.83	2382.97	2917.03	LIMIT	2052.42	2451.62	
		Node	10353	10357	4992	10307	10401	11015	11197	12101	10812	11001	

$J$  increased from 32 to 40). In particular, the CPU time of directly using GUROBI is consistently 1 or 2 orders of magnitude larger than that of our approaches. For smaller  $(I, J)$ -values (e.g.,  $(I, J) = (6, 32)$  or  $(I, J) = (8, 32)$ ), we see that **B&C-Lifted** sometimes runs faster than **B&C-Relax**, but for all the other  $(I, J)$  combinations, the latter completely dominates the former. This is expected because the cuts (21b) are generated in a lifted space with  $J^2$  additional variables for each server  $i = 1, \dots, I$ . Therefore, it makes sense that the scalability of **B&C-Lifted** is worse than that of **B&C-Relax**, which uses cuts (21a) without lifting.

#### 5.2.4 Solving 0-1 SOC models with different $\Lambda$ -values in (11)

We again focus on instances with  $I = 6$  and  $J = 32$  under the same general covariance matrix  $\Sigma$  obtained from the in-sample data points. We let  $\Lambda := \Omega^2 \Sigma$  in the 0-1 SOC constraint (11) and adjust the scalar  $\Omega$  to obtain different  $\Lambda$ . We want to show how the computational time of directly using GUROBI increases as we increase  $\Omega$ , as compared to using the branch-and-cut algorithm with extended polymatroid inequalities. (The results of **B&C-Lifted** are used here and similar observations can be made if the results of **B&C-Relax** are used.)

Considering specific cases of the SOC constraint (11) for modeling DCBP, we have  $\Omega = \Phi^{-1}(1 - \alpha) = 1.64$  for the Gaussian approximation model when  $\alpha = 0.05$ , and  $\Omega = \sqrt{\gamma_2/\alpha} = 5.36$  for DCBP2 when  $\gamma_2 = 2$  and  $\alpha = 0.05$ . We test four values of  $\Omega$  equally distributed in between  $[1.64, 5.36]$  including the two end points.

Figure 2 depicts the average CPU time and the average number of branching nodes of solving five independent DCBP2 instances for each  $\Omega$ -setting. Specifically, the four values 1.64, 2.88,

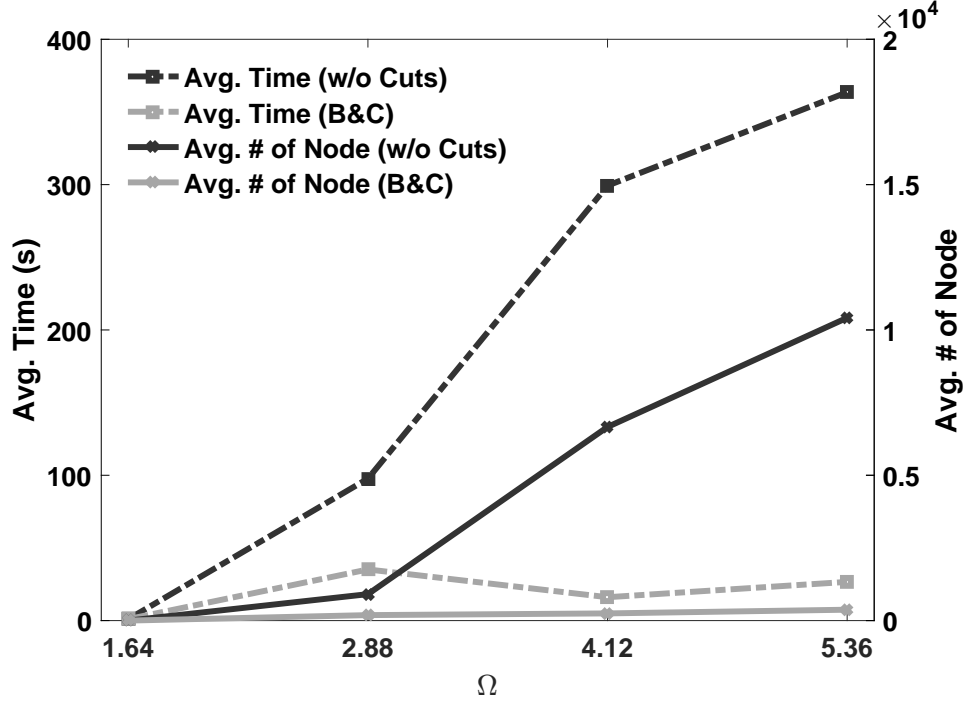


Figure 2: CPU time and number of branching nodes under different  $\Omega$ -values in constraint (11)

4.12, and 5.36 of  $\Omega$  lead to 0.98, 97.68, 299.29, and 363.56 CPU seconds when directly using GUROBI, respectively, together with the significantly growing number of branching nodes 0, 909.8, 6662, and 10418, respectively. On the other hand, the branch-and-cut algorithm with the extended polymatroid inequalities respectively takes 1.03, 35.19, 16.09, and 26.63 CPU seconds on average for solving the same instances, and branches on average 0, 188.6, 247.2, and 374.6 nodes, respectively. This indicates that our approach is more scalable than directly using the off-the-shelf solvers.

### 5.3 Out-of-Sample Performance of DCBP

In this section, we evaluate the out-of-sample performance of the optimal solutions to the DCBP1, DCBP2, Gaussian-based 0-1 SOC models, and the SAA-based MILP model. To generate the out-of-sample reference scenarios, we consider either misspecified distribution type or misspecified moment information as follows.

- Misspecified distribution type:** We sample 10,000 out-of-sample data points from a two-point distribution having the same mean and standard deviation of each random variable  $\tilde{t}_{ij}$  for  $i = 1, \dots, I$  and  $j = 1, \dots, J$  as the in-sample data. The service time is realized as  $\mu_{ij} + \frac{(1-p)}{\sqrt{p(1-p)}}\sigma_{ij}$  with probability  $p$  ( $0 < p < 1$ ) and as  $\mu_{ij} - \frac{\sqrt{p(1-p)}}{(1-p)}\sigma_{ij}$  with probability  $1-p$ , where  $\mu_{ij}$  and  $\sigma_{ij}$  are the sample mean and standard deviation of  $\tilde{t}_{ij}$  obtained from the in-sample data. We set  $p = 0.3$  so that we have smaller probability of having larger service time realizations.



- **Misspecified moments:** Alternatively, we sample 10,000 data points from the Gaussian distribution, but only consider the hM $\ell$ V type of appointments, instead of an equal mixture of all the four types. In each sample and for each  $i = 1, \dots, I$ , we draw a standard-Gaussian random number  $\rho_i$ , and for each  $j = 1, \dots, J$ , generate a service time realization as  $\mu_{ij} + \rho_i \sigma_{ij}$ .

### 5.3.1 Performance of solutions under diagonal matrices

Under diagonal matrices, both of the two DCBP models open three servers (i.e., Servers 4, 5, 6 by DCBP1 and Servers 2, 4, 5 by DCBP2), while the Gaussian and SAA approaches only open Servers 4 and 6. We first use the 10,000 out-of-sample data points given by misspecified distribution type, namely, the two-point distribution. Table 4 reports each solution’s probability of having the total time of assigned appointments not exceeding the capacity of the server to which they are assigned.

Table 4: Solution reliability in out-of-sample data following a misspecified distribution type

Model	Server 2	Server 4	Server 5	Server 6
DCBP1	N/A	1.00	1.00	1.00
DCBP2	1.00	1.00	1.00	N/A
Gaussian	N/A	0.69	N/A	0.91
SAA	N/A	0.69	N/A	0.91

“N/A”: the server is not opened by using the corresponding method.

Recall that  $\alpha_i = 0.05$  for all  $i$  used in all four approaches. The reliability results of the Gaussian and SAA approaches are significantly lower than the desired probability threshold  $1 - \alpha_i = 0.95$  on Server 4, and slightly lower than 0.95 on Server 6. On the other hand, the optimal solutions of DCBP1 and DCBP2 do not exceed the capacity of any open servers.

Next, we use the 10,000 out-of-sample data points given by misspecified moments. Table 5 reports the reliability performance of each optimal solution. The DCBP2 solution still outperforms

Table 5: Solution reliability in out-of-sample scenarios with misspecified moments

Model	Server 2	Server 4	Server 5	Server 6
DCBP1	N/A	0.94	1.00	1.00
DCBP2	0.98	1.00	0.99	N/A
Gaussian	N/A	0.59	N/A	0.89
SAA	N/A	0.59	N/A	0.89

“N/A”: the server is not opened by using the corresponding method.

solutions given by all the other approaches and achieves the desired reliability in all the three open servers. The Gaussian and SAA solutions perform poorly when the moment information is different from the empirical inputs. The DCBP1 solution respects the capacities of Servers 5 and 6 with sufficiently high probability (i.e.,  $> 0.95$ ), but yields a slightly lower reliability (0.94) than the threshold on Server 4.

### 5.3.2 Performance of solutions under general matrices

We optimize all the models under general matrices by using the empirical covariance matrices of the in-sample data, and report their corresponding solutions in Table 6. Each entry illustrates the number of appointments assigned to an open server. Note that the Gaussian and SAA approaches yield the same solution of opening servers and assigning appointments.

Table 6: Optimal open servers and appointment-to-server assignments under general matrices

Model	Server 3	Server 4	Server 5	Server 6
DCBP1	12	N/A	13	7
DCBP2	12	11	9	N/A
Gaussian	15	N/A	17	N/A
SAA	15	N/A	17	N/A

“N/A”: the server is not opened by using the corresponding method.

We test the solutions shown in Table 6 in the out-of-sample scenarios under misspecified distribution type, and present their reliability performance in Table 7. We again show that under general covariance matrices, the DCBP2 model yields the most conservative solution that does not exceed any open server’s capacity, while DCBP1 only ensures the desired reliability on Servers 3 and 6, but not on Server 5. The Gaussian and SAA approaches cannot produce solutions that can achieve the desired reliability threshold on any of their open servers.

Table 7: Solution reliability in out-of-sample scenarios with misspecified moments

Model	Server 3	Server 4	Server 5	Server 6
DCBP1	1.00	N/A	0.91	1.00
DCBP2	1.00	1.00	1.00	N/A
Gaussian	0.91	N/A	0.91	N/A
SAA	0.91	N/A	0.91	N/A

“N/A”: the server is not opened by using the corresponding method.

## 6 Conclusions

In this paper, we considered a DCBP model, where the true distributional information of the random item weights are unknown and only the empirical first and second moments are given. The goal is to minimize the total cost of opening bins and assigning items, while restricting the worst-case probability of exceeding each open bin’s capacity under a given threshold. We provided 0-1 SOC representations of the DCBP model under two types of ambiguity sets. Additionally, we derived extended polymatroid inequalities in both original and lifted space of the 0-1 SOC reformulation. Via extensive numerical studies, we demonstrated that (a) our solution approaches significantly accelerate solving the DCBP model as compared to the state-of-the-art commercial solvers, and (b) DCBP yields reliable out-of-sample performance even when the distributional information is misspecified. We also observed that the branch-and-cut algorithm with extended polymatroid inequalities in the original space scales very well as the problem size grows.

For future research, we plan to apply the SOC representation of DRCC in other application areas, e.g., appointment scheduling, production planning, and power system operation. We also plan to investigate DCBP models under other types of ambiguity sets, which could take into account not only the moment information of item weights but also density or structural information of their probability distribution. The connections between SOC program, SDP, and submodular optimization would also be interesting to study.

## References

- Atamtürk, A. and Narayanan, V. (2008). Polymatroids and mean-risk minimization in discrete optimization. *Operations Research Letters*, 36(5):618–622.
- Bertsimas, D., Doan, X. V., Natarajan, K., and Teo, C.-P. (2010). Models for minimax stochastic linear optimization problems with risk aversion. *Mathematics of Operations Research*, 35(3):580–602.
- Calafiore, G. C. and El Ghaoui, L. (2006). On distributionally robust chance-constrained linear programs. *Journal of Optimization Theory and Applications*, 130(1):1–22.
- Chen, W., Sim, M., Sun, J., and Teo, C. P. (2010). From CVaR to uncertainty set: Implications in joint chance constrained optimization. *Operations Research*, 58(2):470–485.
- Cheng, J., Delage, E., and Lisser, A. (2014). Distributionally robust stochastic knapsack problem. *SIAM Journal on Optimization*, 24(3):1485–1506.
- Delage, E. and Ye, Y. (2010). Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58(3):595–612.
- Deng, Y., Shen, S., and Denton, B. T. (2016). Chance-constrained surgery planning under conditions of limited and ambiguous data. Available at SSRN: <http://dx.doi.org/10.2139/ssrn.2432375>.
- Denton, B. T., Miller, A. J., Balasubramanian, H. J., and Huschka, T. R. (2010). Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, 58(4):802–816.
- Edmonds, J. (1970). Submodular functions, matroids, and certain polyhedra. *Combinatorial Structures and Their Applications*, pages 69–87.
- El Ghaoui, L., Oks, M., and Oustry, F. (2003). Worst-case Value-at-Risk and robust portfolio optimization: A conic programming approach. *Operations Research*, 51(4):543–556.
- Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations Research*, 9(6):849–859.
- Goyal, V. and Ravi, R. (2010). A PTAS for the chance-constrained knapsack problem with random item sizes. *Operations Research Letters*, 38(3):161–164.
- Han, J., Lee, K., Lee, C., Choi, K.-S., and Park, S. (2016). Robust optimization approach for a chance-constrained binary knapsack problem. *Mathematical Programming*, 157(1):277–296.
- Jiang, R. and Guan, Y. (2016). Data-driven chance constrained stochastic program. *Mathematical Programming, Series A*, 158(1):291–327.
- Kleinberg, J., Rabani, Y., and Tardos, É. (2000). Allocating bandwidth for bursty connections. *SIAM Journal on Computing*, 30(1):191–217.
- Kosuch, S. and Lisser, A. (2010). Upper bounds for the 0-1 stochastic knapsack problem and a B&B algorithm. *Annals of Operations Research*, 176(1):77–93.

- Küçükyavuz, S. (2012). On mixing sets arising in chance-constrained programming. *Mathematical Programming*, 132(1–2):31–56.
- Liu, X., Küçükyavuz, S., and Luedtke, J. (2016). Decomposition algorithms for two-stage chance-constrained programs. *Mathematical Programming*, 157(1):219–243.
- Luedtke, J. (2014). A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. *Mathematical Programming*, 146(1–2):219–244.
- Luedtke, J. and Ahmed, S. (2008). A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699.
- Luedtke, J., Ahmed, S., and Nemhauser, G. (2010). An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming*, 122(2):247–272.
- Pagnoncelli, B., Ahmed, S., and Shapiro, A. (2009). Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, 142(2):399–416.
- Prékopa, A. (1970). On probabilistic constrained programming. In *Proceedings of the Princeton Symposium on Mathematical Programming*, pages 113–138.
- Prékopa, A. (2003). Probabilistic programming. In *Stochastic Programming: Handbooks in Operations Research and Management Science*, volume 10, chapter 5, pages 267–351. Elsevier.
- Reich, D., Shi, Y., Epelman, M., Cohn, A., Barnes, E., Arthurs, K., and Klampfl, E. (2015). Scheduling crash tests at Ford Motor Company. To appear in *Interfaces*.
- Scarf, H., Arrow, K., and Karlin, S. (1958). A min-max solution of an inventory problem. In *Studies in the Mathematical Theory of Inventory and Production*, volume 10, pages 201–209. Stanford University Press, Stanford, CA.
- Shen, S. and Wang, J. (2014). Stochastic modeling and approaches for managing energy footprints in cloud computing services. *Service Science*, 6(1):15–33.
- Shylo, O. V., Prokopyev, O. A., and Schaefer, A. J. (2012). Stochastic operating room scheduling for high-volume specialties under block booking. *INFORMS Journal on Computing*, 25(4):682–692.
- Simchi-Levi, D., Chen, X., and Bramel, J. (2014). *The Logic of Logistics: Theory, Algorithms, and Applications for Logistics Management*. Springer Verlag.
- Smith, J. E. and Winkler, R. L. (2006). The optimizer’s curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322.
- Song, Y., Luedtke, J. R., and Küçükyavuz, S. (2014). Chance-constrained binary packing problems. *INFORMS Journal on Computing*, 26(4):735–747.
- Wagner, M. (2008). Stochastic 0–1 linear programming under limited distributional information. *Operations Research Letters*, 36(2):150–156.
- Wang, J. and Shen, S. (2012). Risk and energy consumption tradeoffs in cloud computing service via stochastic optimization models. In *Proceedings of the 5th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2012)*, Chicago, IL.
- Wiesemann, W., Kuhn, D., and Sim, M. (2014). Distributionally robust convex optimization. *Operations Research*, 62(6):1358–1376.
- Zhang, Z., Denton, B., and Xie, X. (2015). Branch and price for chance constrained bin packing. Available at Optimization-Online [http://www.optimization-online.org/DB\\_HTML/2015/11/5217.html](http://www.optimization-online.org/DB_HTML/2015/11/5217.html).

Zymler, S., Kuhn, D., and Rustem, B. (2013). Distributionally robust joint chance constraints with second-order moment information. *Mathematical Programming*, 137(1-2):167–198.